We claim:

1. A method for scheduling program units, the method comprising:

starting a process within an operating system;

starting at least one thread within the operating system, the thread associated with the process;

executing a plurality of streams within the thread;

entering a kernel mode by a first stream of the plurality of streams upon the occurrence of a context shifting event;

if the first stream must block, then blocking the execution of the other streams of the plurality of streams.

2. The method of claim 1, further comprising saving the context of each of the plurality of streams in a thread context data structure.

3. The method of claim 2, wherein each of the streams are executed on a separate processor.

4. The method of claim 1, wherein the context shifting event comprises an exception.

5. The method of claim 4 wherein the exception comprises a signal.

6. The method of claim 1 wherein the context shifting event comprises a non-local goto.

7. The method of claim 1, wherein the context shifting event comprises a system call.

8. A system for scheduling streams, the system comprising:

at least one multiple processor unit having a plurality of processors;

a memory coupled to the plurality of processors; and

an operating environment executed by at least one of the processors from the memory and operable to perform the tasks of:

start a process within an operating system,

start at least one thread within the operating system, the thread associated with the process;

execute a plurality of streams within the thread,

enter a kernel mode by a first stream of the plurality of streams upon the occurrence of a context shifting event, and

if the first stream must block, then blocking the execution of the other streams of the plurality of streams.

9. The system of claim 8, further comprising saving the context of each of the plurality of streams in a thread context data structure.

10. The system of claim 9, wherein each of the streams are executed on a separate processor of the multiple processor unit.

11. The system of claim 8, wherein the context shifting event comprises an exception.

12. The system of claim 11 wherein the exception comprises a signal.

13. The system of claim 8 wherein the context shifting event comprises a non-local goto.

14. The system of claim 8, wherein the context shifting event comprises a system call.

15. A computer-readable media having computer executable instructions for performing a method for scheduling program units, the method comprising:

starting a process within an operating system;

starting at least one thread within the operating system, the thread associated with the process;

executing a plurality of streams within the thread;

entering a kernel mode by a first stream of the plurality of streams upon the occurrence of a context shifting event;

if the first stream must block, then blocking the execution of the other streams of the plurality of streams.

16. The computer-readable media of claim 15, further comprising saving the context of each of the plurality of streams in a thread context data structure.

17. The computer-readable media of claim 16, wherein each of the streams are executed on a separate processor.

18. The computer-readable media of claim 15, wherein the context shifting event comprises an exception.

19. The computer-readable media of claim 18 wherein the exception comprises a signal.

20. The computer-readable media of claim 15 wherein the context shifting event comprises a non-local goto.

21. The computer-readable media of claim 15, wherein the context shifting event comprises a system call.